Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (canceled)

2. (currently amended) The method of claim ~~1~~7, wherein during execution the deferred servicing procedure services all pending interrupts from all of the plurality of co-processors.

3. (currently amended) The method of claim ~~1~~7, wherein the plurality of pending interrupts serviced by the deferred servicing procedure includes a second interrupt generated by the one of the plurality of co-processors that generated the detected interrupt.

4. (currently amended) The method of claim ~~1~~7, wherein the plurality of pending interrupts serviced by the deferred servicing procedure includes a second interrupt generated by one of the plurality of co-processors other than the one that generated the detected interrupt.

5. (canceled)

6. (currently amended) The method of claim ~~5~~7, wherein in the event that the value stored in the interrupt register does not indicate an interrupt, a different one of the co-processors is selected and the act of reading is repeated.

7. (currently amended) ~~The method of claim 5,~~ A method for servicing interrupts generated by a plurality of co-processors included in a multiprocessor subsystem, the method comprising the acts of:

in response to a detected interrupt, determining whether the detected interrupt was generated by one of the plurality of co-processors of the multiprocessor subsystem; and

in the event that the detected interrupt was generated by one of the plurality of co-processors, scheduling execution of a deferred servicing procedure,

wherein during execution the deferred servicing procedure services a plurality of pending interrupts generated by two or more of the plurality of co-processors, including the detected interrupt,

wherein the act of determining whether the detected interrupt was generated by one of the plurality of co-processors includes the acts of:

selecting one of the plurality of co-processors as a current co-processor; and

reading a value stored in an interrupt register of the current co-processor, and

wherein the act of reading includes:

updating a private register mapping to enable access to the interrupt register of the current co-processor,

wherein the private mapping is not used by the deferred servicing procedure.

8. (currently amended) The method of claim ~~1~~7, further comprising the act of disabling further interrupts from the plurality of co-processors in the event that the detected interrupt was generated by one of the plurality of co-processors,

wherein during execution the deferred servicing procedure re-enables interrupts from the plurality of co-processors.

9. (original) The method of claim 8, wherein the act of disabling further interrupts is performed at a critical priority level.

10. (currently amended) The method of claim ~~1~~7, wherein the act of determining whether the detected interrupt was generated by one of the plurality of co-processors is performed at a critical priority level.

11. (original) The method of claim 10, wherein the act of scheduling execution of the deferred servicing procedure is performed at the critical priority level.

12. (original) The method of claim 11, wherein the act of scheduling execution of the deferred servicing procedure includes setting a second priority level for the deferred servicing procedure, wherein the second priority level is lower than the critical priority level.

13. (currently amended) The method of claim ~~1~~7, wherein the multiprocessor subsystem is a graphics processing subsystem.

14. (canceled)

15. (currently amended) The system of claim ~~14~~20 wherein the interrupt detection module is further configured to be activated by a central processing unit of the computer system in response to an interrupt signal.

16. (currently amended) The system of claim ~~14~~20, wherein the interrupt detection module is further configured to disable further interrupts from all of the co-processors in the event of an interrupt from any one of the co-processors, and wherein the servicing module is further configured to re-enable further interrupts from all of the co-processors.

17. (currently amended) The system of claim ~~14~~20, wherein the interrupt detection module is further configured to operate at a critical priority level and the servicing module is further configured to operate at a second priority level lower than the critical priority level.

18. (currently amended) The system of claim ~~14~~20, wherein the multiprocessor subsystem is configured for graphics processing.

19. (canceled)

20. (currently amended) ~~The system of claim 19,~~ A computer system comprising:

a multiprocessor subsystem including a plurality of co-processors for processing data, wherein each of the co-processors is configured to generate interrupts; and

a driver module configured to control operation of the multiprocessor subsystem, the driver module including:

a schedulable servicing module configured to detect and service all pending interrupts from all of the co-processors when activated; and

an interrupt detection module configured to schedule the servicing module for activation in the event of an interrupt from any one of the plurality of co-processors,

wherein each of the plurality of co-processors includes an interrupt register configured to indicate an interrupt, and wherein the interrupt detection module is further configured to determine whether one of the plurality of co-processors generated an interrupt by accessing the respective interrupt registers of the co-processors,

wherein the interrupt detection module is further configured to maintain a private mapping for accessing the respective interrupt registers, the private mapping being used exclusively by the interrupt detection module.

21. (currently amended) A computer program product comprising~~:~~ a computer readable storage medium encoded with program code, the program code including:

program code for determining, in response to a detected interrupt, whether the detected interrupt was generated by one of the plurality of co-processors of the multiprocessor subsystem;

program code for scheduling a deferred servicing procedure in the event that the detected interrupt was generated by one of the plurality of co-processors; and

program code for performing the deferred servicing procedure, wherein the program code for performing the deferred servicing procedure includes program code for servicing a plurality of pending interrupts from two or more of the plurality of processors, including the detected interrupt,

wherein the program code for determining whether the detected interrupt was generated by one of the plurality of co-processors includes program code for:

selecting one of the plurality of co-processors as a current co-processor; and

reading a value stored in an interrupt register of the current co-processor, and

wherein the program code for reading includes:

program code for updating a private register mapping to enable access to the interrupt register of the current co-processor, wherein the private mapping is not used by the deferred servicing procedure.

22. (currently amended) The method of claim ~~1~~7, further comprising:

determining whether interrupts from the plurality of co-processors are enabled; and

if the interrupts from the plurality of co-processors are not enabled, exiting without performing further processing.

23. (previously presented) The method of claim 2, wherein servicing all pending interrupts from all of the plurality of co-processors comprises:

loading by the deferred servicing procedure a mapping to the registers of a next co-processor to be serviced; and

servicing any and all pending interrupts stored in the registers of the next co-processor.